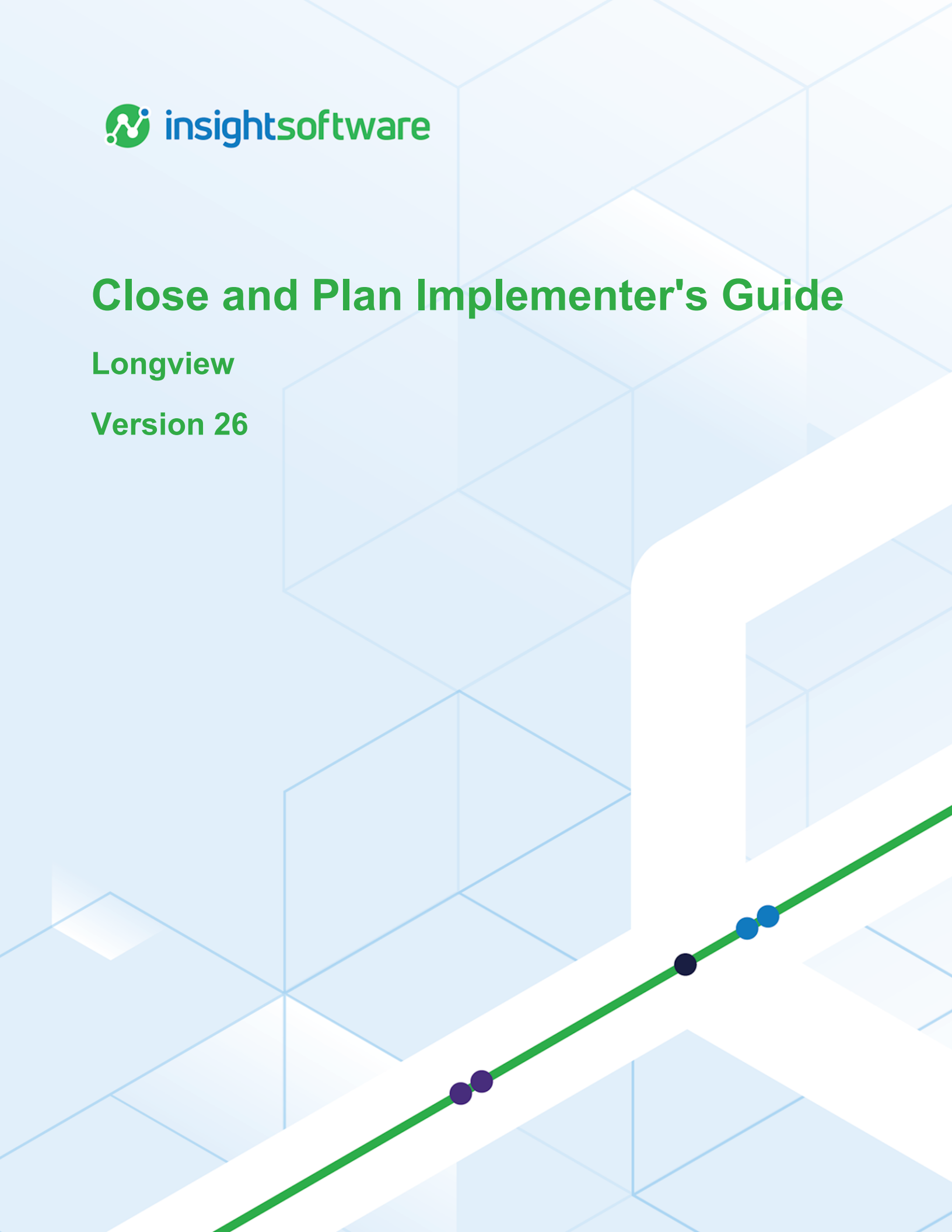
insightsoftware

Close and Plan Implementer's Guide

Longview

Version 26



Document Information

Notices

Copyright

The information contained in this document represents the current view of insightsoftware.com on the issues discussed as of the date of publication. This document is for informational purposes only. insightsoftware.com makes no representation, guarantee or warranty, expressed or implied, that the content of this document is accurate, complete or up to date.

Disclaimer

This guide is designed to help you to use the Longview applications effectively and efficiently. All data shown in graphics are provided as examples only. The example companies and calculations herein are fictitious. No association with any real company or organization is intended or should be inferred.

Contents

Introduction to Longview

Longview provides corporate performance management (CPM) software that leading companies use to drive performance with speed, visibility, and financial integrity. Since 1994, many of the world's most respected companies have been using our technology platform to create a single repository of financial truth from which statutory consolidation, management reporting, financial planning, modeling, analysis, budgeting, forecasting, and strategic tax can be performed quickly and accurately, enterprise wide.

Longview enables enterprise clients to collect, store, analyze, and report on data in real-time by automating, centralizing, and standardizing any one or combination of the following key financial processes: Planning, Budgeting, Forecasting, Consolidation, Financial Close Management, Profitability Analytics, Statutory, XBRL Financial Reporting and Tax Provisioning. With Longview customers can reduce overreliance on spreadsheets, improve transparency, and regain control of these key finance functions.

Longview Tax calculates your company's global tax charge, effective tax rate, and deferred taxes for tax provisioning purposes. Since Longview Tax uses the same technological platform as your corporate performance management solution, the tax reporting process is directly integrated into the corporate close process. As one solution, consolidated pre-tax income can be reported by legal entity to accurately calculate consolidated income tax charges and deferred taxes.

For more information on purchasing Longview Tax, contact your Longview Account Manager. Web services are a standardized way of integrating applications over the Internet or Internet protocol-based networks. Web services rely on certain software standards including Extensible Markup Language (XML), Simple Object Access Protocol (SOAP), Web Service Definition Language (WSDL) and Universal Description, Discovery & Integration (UDDI).

About this guide

This guide includes basic information on how to install your Longview system. The following sections indicate conventions that are used in this guide.

Warnings and notes

This guide uses the following conventions for warnings and notes:

Caution: Warnings provide cautionary information on the possible effect of certain actions, including the unintentional deletion of data. Be sure to read and understand all warnings before performing a related procedure.

Note: Notes provide additional information to help you understand your Longview system better. They also provide important information on exceptions to general guidelines.

Procedures

There may be several ways to perform a procedure in your Longview system.

- You may be able to choose a menu command. For example, to open a file, you can choose **Open** from the File menu. In this documentation, we use: Choose **File** > **Open**.

- You may be able to use a keyboard equivalent. For example, to exit you can press the **Alt** key and then the letter **F** to open the File menu, and then press **X** for Exit.
- You may be able to click a button or icon. If a menu command has an equivalent button or icon, an illustration of the button or icon may appear in the margin.

In this documentation, we may not describe all methods to carry out a task. Use whichever method you prefer. Depending on the task you are performing, certain methods may not be available.

Contacting Longview

Questions? We are ready to help. For contact information for Longview, visit our web site at insightsoftware.com/Longview/.

Close and Plan Implementation Overview

Once the initial installation and setup are complete, there are several tasks that are required to finish the initial setup and implementation to get the Longview system in a working state. This section is a summary of the tasks required and will refer you to the full documentation in the *Solutions Administrator Guide*.

To access the Solutions Framework Administrator guide:

1. From within the Longview Client, select the **Documentation** module.
2. Select the **Solutions Framework Administrator Guide**. The guide appears in your browser.

To access the Close And Plan Administrator guide:

1. From within the Longview Client, select the **Documentation** module.
2. Select the **Close And Plan Administrator Guide**. The guide appears in your browser.

This section includes the following topics:

Initial Implementation Steps

For a successful initial implementation, follow these steps in order:

1. Adding currencies to the system
2. Adding currency translations to the system
3. Adding time periods to the system
4. Complete time period setup
5. Update server configuration
6. Set up trial balance accounts
7. Set up consolidation structures
8. Set up intercompany eliminations
9. Enable intercompany eliminations
10. Set up roll forward accounts
11. Set up cash flow accounts
12. Set up custom dimensions
13. Update rules for custom dimensions
14. Configure categories

Adding currencies to the system

Currencies must be added before entities and other settings to allow for the configuration of currency related settings, such as the functional currency for an entity.

Note: For more details See “Managing currencies” in the *Close And Plan Administrator Guide* .

To add currencies to the system:

1. Create a "csv" file containing the currencies to be created (including the header).

```
Translate,@Budget Rates,@Prior Year Rates,Currency
0,0,0,"CAD"
1,0,0,"USD"
```

2. Select the **Consolidate** module.
3. Select the **Administration** category.
4. Expand **Currencies**.

5. Select **Manage Currencies**. The Manage Currencies dialog appears.
6. Choose the **Import** option.
7. Click the **file browser** and select the **.csv file** created in step 1.
8. Click **OK**.
9. Review the results.

Adding currency translations to the system

Currency translations are added to generate reporting results in the correct currency.

Note: For more details, see “Managing translations” in the *Close And Plan Administrator Guide*.

To add currency translations to the system:

1. Create a "csv" file containing the translations to be created (including the header).

```
Source,Target,Method
"CAD","USD","TASC"
```

2. Select the **Consolidate** module.
3. Select the **Administration** category.
4. Expand **Currencies**.
5. Select **Manage Translations**. The Manage Translations dialog appears.
6. Choose the **Import** option.
7. Click the **file browser** and select the **.csv file** created in step 1.
8. Click **OK**.
9. Review the results.

Adding time periods to the system

Time periods must be added to the system before continuing.

Note: For more details, see “Rollover, create year” in the *Close And Plan Administrator Guide*.

To add time periods to the system:

1. Select the **Consolidate** module.
2. Select the **Administration** category.

3. Expand **Time Periods**.
4. Select **Create Year**.
5. Enter the earliest year required in the system for initial setup.
6. Select **Actual**.
7. Click **OK**.
8. Repeat for each subsequent year, selecting Actual, Forecast and Budget options as required.

Complete time period setup

Time period setup is finalized by setting the current year and period in the system.

> **Note:** For more details, see “System setup, time period setup” in the *Close And Plan Administrator Guide*.

To complete time period setup:

1. Select the **Consolidate** module.
2. Select the **Administration** category.
3. Expand **System**.
4. Select **Time Period Setup**.
5. Select the **current year**.
6. Select the **current period**.
7. Click **OK**.

Update server configuration

At this point the server configuration should be updated and the servers restarted to apply the changes.

To update server configuration:

1. Close all open applications.
2. Launch Server Manager.
3. Right-click on the **server** and select **Server Configuration**.
4. Select **Application**.
5. Check **Period Activity Calculation (PAC)**.
6. Check **Retained Earnings Calculation (REC)**.
7. Check **Foreign Exchange (TRN)**.
8. Check **Year to Date Calculation (YTD)**.

9. Check **Journal Entries**.
10. Click **OK**.
11. Restart the servers.

Set up trial balance accounts

Trial balance accounts are added next as subsequent configuration requires these accounts.

> **Note:** For more details, see “Managing trial balance accounts” in the *Close And Plan Administrator Guide*.

To set up trial balance accounts:

1. Create a "csv" file containing the trial balance accounts to be created (including the header).

```
Name, Description, Balance Type, Parent, Sort Order, Foreign Exchange Treatment
"TA2020","Trade and other current
receivables","DEBIT","TA2000",2020.000000000,"NULL"
"11200","Trade receivables","DEBIT","TA2020",11200.000000000,"Monetary"
"11209","Trade receivables,
provision","DEBIT","TA2020",11209.000000000,"Monetary"
```

2. Select the **Consolidate** module.
3. Select the **Administration** category.
4. Expand **Accounts**.
5. Select **Manage Trial Balance**. The Manage Trial Balance dialog appears.
6. Choose the **Import** option.
7. Click the **file browser** and select the **.csv file** created in step 1.
8. Click **OK**.
9. Review the results.

Set up consolidation structures

Before setting up legal entities, consolidation structures must be created.

> **Note:** For more details, see “Entities, Managing Consolidations” in the *Close And Plan Administrator Guide*.

To set up consolidation structures:

1. Create a "csv" file containing the consolidation entities to be created (including the header).

```
Entity,Description,Rollup Type, Functional Currency,Parent Entity,Sort Order
"E10000C","ABC Holding, Consolidated","C","USD","NULL",0.000000000
"E11000C","Alpha Holding, Consolidated","C","USD","E10000C",11000.000000000
"E12000C","Beta Holding, Consolidated","C","USD","E10000C",12000.000000000
```

2. Select the **Consolidate** module.
3. Select the **Administration** category.
4. Expand **Entities**.
5. Select **Manage Consolidations**. The Manage Consolidation dialog appears.
6. Choose the **Import** option.
7. Click the **file browser** and select the **.csv file** created in step 1.
8. Click **OK**.
9. Review the results.

Set up legal entities

Once the consolidation structures have been set up, legal entities can be added to the system.

> **Note:** For more details, see “Entities, Managing Entities” in the *Close And Plan Administrator Guide*.

To set up legal entities:

1. Create a "csv" file containing the entities to be created (including the header)

```
Entity,Description,Functional Currency,Parent Entity,Sort Order,Target EQP
Entity,Target NCI Entity,Target Balance Sheet Account, Target Income Statement
Account
"E11110","Toronto","CAD",[E11000C],11110.000000000,[NULL],[NULL],[NULL],[NULL]
"E12140","Vancouver","CAD",[E12000C],12140.000000000,[NULL],[NULL],[NULL],
[NULL]
"E11140","Zurich","CHF",[E11000C],11140.000000000,[NULL],[NULL],[NULL],[NULL]
"E12130","Munich","EUR",[E12000C],12130.000000000,[NULL],[NULL],[NULL],[NULL]
"E11150","Equity PickupCo.","GBP",[NULL],0.000000000,[E11130],[NULL],[16441],
[81060]
"E13100","Singapore","SGD",[E10000C],13100.000000000,[NULL],[E10000CA],[NCI_
CY],[NCI_NI]
```

2. Select the **Consolidate** module.
3. Select the **Administration** category.
4. Expand **Entities**.
5. Select **Manage Entities**. The Manage Entities dialog appears.
6. Choose the **Import** option.
7. Click the **file browser** and select the **.csv file** created in step 1.
8. Click **OK**.
9. Review the results.

Set up intercompany eliminations

Now that currencies, accounts and entities have been set up, intercompany eliminations can be configured.

To set up intercompany eliminations:

1. Launch Application Administrator.
2. Select **Intercompany Eliminations**.
3. Add standard and investment eliminations, as required.
4. Turn maintenance mode off by selecting **Actions** > **Maintenance** in the menu.

Enable intercompany eliminations

Now that intercompany eliminations have been configured, the server configuration needs to be updated and the servers restarted.

To enable intercompany eliminations:

1. Close all open applications.
2. Launch Server Manager.
3. Right-click on the **server** and select **Server Configuration**.
4. Select **Application**.
5. Check **Intercompany Eliminations (Elim)**.
6. Click **OK**.
7. Restart the servers.

Set up roll forward accounts

Now that trial balance accounts are configured, roll forward accounts can be set up.

Note: For more details, see “Roll Forward, Managing roll forward accounts” in the *Close And Plan Administrator Guide*.

To set up roll forward accounts:

1. Create a "csv" file containing the roll forward accounts to be created (including the header).

```
Source Account, Roll Forward Account, Description, Acquisitions, Additions,
Disposals, Sale of Assets, Repayments, Parent Account, Sort Order

[NULL],"RF_Net_FA","Net Fixed Assets",0,0,0,0,0,"DATAVIEWS_RF",0.000000000

[NULL],"RF_Building","Building",0,0,0,0,0,"RF_Net_FA",10.000000000

[14110],"RF_14110","Fixed Assets - Building",1,1,0,1,0,"RF_
Building",0.000000000

[14210],"RF_14210","Acc. Dep - Building",0,0,1,0,0,"RF_Building",0.000000000
```

2. Select the **Consolidate** module.
3. Select the **Administration** category.
4. Expand **Accounts**.
5. Select **Manage Roll Forwards**. The Manage Roll Forwards dialog appears.
6. Choose the **Import** option.
7. Click the **file browser** and select the **.csv file** created in step 1.
8. Click **OK**.
9. Review the results.

Set up cash flow accounts

Note: For more details, see “Managing cash flow accounts” in the *Close And Plan Administrator Guide*.

Now that trial balance and roll forward accounts are configured; cash flow accounts can be set up.

To set up cash flow accounts:

1. Create a "csv" file containing the cash flow accounts to be created (including the header).

```
Account,Description,Source Account,Source Roll Forward,Impact,Calculate on
Source and Translated Values,Parent Account,Sort Order

"TCF1000","Cash flows from (used in) operating activities",[NULL],
[NULL],"NULL",0,"TCF4000",1000.000000000
```

```
"TCF2000","Cash flows from (used in) investing activities",[NULL],
[NULL],"NULL",0,"TCF4000",2000.000000000
"TCF3000","Cash flows from (used in) financing activities",[NULL],
[NULL],"NULL",0,"TCF4000",3000.000000000
```

2. Select the **Consolidate** module.
3. Select the **Administration** category.
4. Expand **Accounts**.
5. Select **Manage Cash Flow**. The Manage Cash Flow dialog appears.
6. Choose the **Import** option.
7. Click the **file browser** and select the **.csv file** created in step 1.
8. Click **OK**.
9. Review the results.

Set up custom dimensions

By default, the system assumes that trial balance will be detailed by the “Total” symbol in each custom dimension. If this is not the case, update the appropriate system attributes to configure the symbols to use. This includes cash flow source and target attributes, intercompany input attributes and non-controlling interests target attributes.

Note: For more details, see “Managing hierarchies” in the *Close And Plan Administrator Guide*.

To set up hierarchies in custom dimensions:

1. Create a "csv" file containing the symbols to be created (including the header).

```
Name,Description,Type,Parent,Sort Order,Weight
"PHW","Hardware","STANDARD","PRODUCTS_Total",10.000000000,"+"
"PHW001","CPU","STANDARD","PHW",10.000000000,"+"
```

2. Select the **Consolidate** module.
3. Select the **Administration** category.
4. Expand **Other Dimensions**.
5. Select **Manage Hierarchies**. The Manage Hierarchies dialog appears.
6. Select the **dimension** and **root symbol** to manage.
7. Click **OK**.

8. At the next dialog, choose the **Import** option.
9. Click the **file browser** and select the **.csv file** created in step 1.
10. Click **OK**.
11. Review the results.
12. Repeat for each custom dimension.

Update rules for custom dimensions

In cases where custom dimensions deviate from the assumed standard or standard accounts are modified, validation and event rules need to be updated.

The table below outlines the rules to be updated. As an alternative, you can delete the rules and run the Install process again, which will re-create the rules using the updated cash flow source and target attributes. You will need to apply changes to the CASH account manually.

Related Cash Flow Attribute	Rule
SOURCE (ASCFDxSource)	1000010, 1100010, 1200010
	1000030, 1100030, 1200030
	1000050
	1000060, 1100060, 1200060
	5000011, 5100011, 5200011 (CASH)
	5000021, 5100021, 5200021 (CASH)
	5000031, 5100031, 5200031
	5000061
	5000071
	5000076
	5000091
	5000101
SOURCE (ASCFDxSource) and TARGET (ASCFDxTarget)	1000020, 1100020, 1200020 (CASH)
TARGET (ASCFDxTarget)	1000040
	5000051, 5100051, 5200051
	5000081

Configure categories

The standard module files in the Longview Client are divided into:

- Data Collection
- Reports
- Workflow
- Administration

The folders within Data Collection and Reports are defined by categories. These categories can be configured with up to ten active categories in each area. On initial install only a few of these are activated to hold the out of box items.

To configure categories:

1. Select the **Design** module.
2. Select the **Administration** category.
3. Click **Configure Categories**.
4. Activate additional categories, as required.

Working With Examples

Solutions framework includes several example apps built within the free form app template to help kick start implementation of commonly configured functionality. On initial installation, these example apps will be included and can be accessed in the Design module of the client.

Note: Examples are not fully functional apps, but are intended to show examples of common functionality within the framework.

Solutions Framework Free Form examples

The example apps are included and updated in each release. The apps are intended as examples to be duplicated or copied from, but not modified to run as subsequent releases may include modifications.

To review an example app:

1. Select the **Design** module.
2. Select the **Apps** category.
3. Expand **Configuration**.
4. Expand **Free Form**.
5. Expand **Existing Apps**.
6. Click in the **app** you wish to view, or right-click on the **app** you wish to start with and select **Duplicate** to make a copy of it.

The following sample apps are included:

- EXAMPLE - Framework app
- EXAMPLE - Import app

EXAMPLE - Framework app

The example framework app provides examples of using framework utilities described in the solutions developer guide. The example contains a procedure for each code library within the framework with sample code for each utility.

Execute Procedure

The execute procedure is intentionally blank, as this example app is not intended to be run, but as a quick reference for using solutions framework utilities.

Procedures\CORE

The CORE procedure contains sample code for using the following CORE utilities:

1. CreatePatternsLVDSP: Used to generate the data spec document to retrieve patterns from the database.
2. GetDimensionIndex: Used to get the index of the defined dimension.

3. Rollup: Used to perform a full rollup on a data area.
4. GetKeyDimensions: Used to populate the variable CORE_KeyDimensionList with the list of key dimensions.
5. ViewFile: Used to open the specified file in the default program configured on the user's computer.

Procedures\DATA

The DATA procedure contains sample code for using the following DATA utilities:

1. CreateTBLVDSP: Used to generate a data spec for custom dimensions to query the trial balance. The document generated needs to be appended to an existing data spec for the standard dimensions. The following specs are created:
   a. TBSrc: Contains specifications for each custom dimension as defined by attribute ASCFD<?>Source with the hierarchical specification: #0
   b. TBTrg: Contains specifications for each custom dimension as defined by attribute ASCFD<?>Target with the hierarchical specification: #99
   c. TBSrcDetail: Contains specifications for each custom dimension as defined by attribute ASCFD<?>Source with the hierarchical specification: #0
   d. TBSrcLeaf: Contains specifications for each custom dimension as defined by attribute ASCFD<?>Source with the hierarchical specification: ###
2. ParseDsp: Used to generate the body of a data report input file from the specified data spec.

Procedures\DEBUG

The DEBUG procedure contains sample code for using the following DEBUG utilities:

1. Message: Used to show or write a message only when troubleshooting mode is enabled.
2. HistoryOff/HistoryOn: Used to temporarily suspend writing to the history log file when troubleshooting mode is enabled and detailed history logging is on.
3. TimerStart/TimerStop: Used to measure the time to execute a series of commands when troubleshooting mode is enabled.

   Note: Detailed history logging must be enabled.

4. VariableList: Used to write the value of one or more variables to the log file when troubleshooting mode is enabled.
5. ExportDataArea: Used to export the values in one or more specified data areas to a file.

   Note: When used in an event the exported data file is appended to the log.

6. WriteDocument: Used to append a dynamic document to the log file when troubleshooting mode is enabled.

Procedures\FORM

The FORM procedure contains sample code for using the following DEBUG utilities:

1. SelectSymbols: Used to generate and display a form to select symbols from one or more dimensions.
2. GenerateSymbolSelectors: Used to generate a document named SymbolSelectors that can be used to append symbol selectors to a dynamic form document.

EXAMPLE - Import app

The example import app provides a quick way to implement an import processing using the IMP code library. The example contains sample procedure code for both Full and Incremental data import methods.

For more details on the IMP code library, see “IMP code library” in the *Longview Developer’s Guide*.

Note: This example assumes that a single data area spec named Main and a single import spec named Main will be used.

Execute Procedure

The execute procedure performs the following steps:

1. Initializes IMP code library.
2. Sets the name of the file to be imported in the IMP_FileName variable.

   Note:
   - This is not required if the data source will be a SQL query executed via ODBC.
   - This variable can also be set via the ShowFileChooser command or a form.
3. Checks that the specified file exists and throws an error if it does not.
4. Locks the target data area.
5. Creates the target data area in memory.

Procedures\FullImport

The FullImport procedure can be found in the Procedures folder and provides the rest of the commands required to execute a full import. Full import is used when the data source will replace the data in the target data area.

Note: Full import does not download the target data area and uses the RECONCILE Upload command. A user submission with status No Data will be created if there are no data changes.

Procedures\IncrementalImport

The IncrementalImport procedure can be found in the Procedures folder and provides the rest of the commands required to execute an incremental import. Incremental import is used when the data source will be added to the data in the target data area.

Note: Incremental import downloads the target data area and uses the Upload command. No user submission will be created if there are no changes to the data.

Procedures\Import

The Import procedure can be found in the Procedures folder and contains the actual Run Import command. The Import procedure is called by the IMP code library procedure OnImport.lvpro.

Close and Plan Data View Input Examples

The example apps are included and updated in each release. The apps are intended as examples to be duplicated or copied from, but not modified as subsequent releases may include modifications.

To review an example app:

1. Select the **Design** module.
2. Select the **Apps** category.
3. Expand **Data Collection**.
4. Expand **Data View Input**.
5. Expand **Existing Apps**.
6. Click on the **app** you wish to view, or right-click on the **app** you wish to start with and select **Duplicate** to make a copy of it.
7. Click the **Customize** button to make modifications. Any changes you make will be saved in the custom files location of your server.

The following sample apps are included:

- Acquisition Balance Sheet app
  - See Common Configuration.
- Cash Flow Adjustment app
  - See Common Configuration, and Specific configuration – Cash Flow Adjustment.
- Discontinued Operations Adjustments app
  - See Common Configuration, and Specific configuration – Discontinued Operations Adjustments.
- Supplemental Input app
  - See Common Configuration.
- Trial Balance Input app
  - See Common Configuration.

Common configuration

Each of the sample apps is based on the data view input template and shared common configuration settings.

Updating the User Selections

The user selections need to be updated to include the corresponding dimension names in your system.

To update the user selections for non-standard dimension names:

1. Expand **User Selections** and double-click on **Dimension 1**.
2. Select the **dimension** that corresponds to TIMEPERIODS in your system. For example, select TIMEPER if that is the name of the dimension in your system.
3. Expand **User Selections** and double-click on **Dimension 2**.
4. Select the **dimension** that corresponds to ENTITIES in your system. For example, select ENTITY if that is the name of the dimension in your system.
5. Click **Save**.

Updating the Attribute Based Selection

The Attribute Based Selections need to be updated to include the corresponding dimension names in your system.

To update the attribute-based selections for non-standard dimension names:

1. Expand **Attribute Based Selections** and double-click on **Dimension 1**.
2. Select the **dimension** that corresponds to CURRENCIES in your system. For example, select CURRENCY if that is the name of the dimension in your system.
3. Select the **dimension** that corresponds to ENTITIES in your system. For example, select ENTITY if that is the name of the dimension in your system.
4. Click **Save**.

Updating the Data Area Definitions

The data area definitions need to be updated to include symbols for all dimensions in your system.

To update the data area definitions:

1. Expand **Data Area Definitions**.
2. For each data area definition listed, select one symbol for each additional dimension. The standard dimensions are ACCOUNTS, TIMEPERIODS, ENTITIES, DATAVIEWS, SCENARIOS and CURRENCIES.
3. Click **Save**.

Updating the Data Area Views

The data area views need to be updated to include symbols for all dimensions in your system.

To update the data area views:

1. Expand **Data Area Views**.
2. For each data area view listed:
   a. Select a **symbol** for each additional dimension. This can be done by using the option “Copy Default Selections” and select from Supplemental.

> **Note:** The expansion level gets reset to 0 when the copy option is used.

b. Expand the additional configuration section and update functions that include area specifications (semi-colon delimited symbols specifications). These functions include:

- ConditionalCellStyle
- NumericInputOnly
- Protect

3. Click **Save**.

Updating Validations

Validations need to be updated to include symbols for all dimensions in your system

To update the validations:

1. Double-click on **Validations**.
2. Update the value statement to include the additional dimensions. For example, if your system has one additional dimension called PRODUCTS, the value command would need to be updated to include the products dimension.
3. Click **Save**.

Specific configuration – Discontinued Operations Adjustment

Updating the Pre-Selection for additional data view symbols

The Pre-Selections needs to be updated to include any additional adjustments or calculated symbols to the variables in this section.

To update the pre-selection for additional data view symbols:

1. Double-click on **Pre-Selection**.
2. Update the dataviews to hide range variables (DVToHide) to include any additional data view symbols you would like to hide from the view. Use a pipe symbol to separate the symbol names (e.g. CF_Calc2|CFFX2)
3. Update the adjustment range variables (DVCFAdj) to include any additional adjustment data view symbols.
4. Click **Save**.

Specific configuration – Cash Flow Adjustment

Updating the Pre-Selection for additional data view or account symbols

The Pre-Selections needs to be updated to include any additional adjustments or calculated symbols to the variables in this section.

To update the pre-selection for additional data view or account symbols:

1. Double-click on **Pre-Selection**.
2. Update the adjustment range variable for adjustments within management reporting. (rDVTotCFAdj). Use a pipe symbol to separate the symbol names (e.g. CF_Reclass|CF_NonCash). This variable contains the list of symbols for cash flow adjustments within management reporting.
3. Update the adjustment range variable for management reporting. (rDVTotCFAdj2). Use a pipe symbol to separate the symbol names (e.g. tCF_Reclass|tCF_NonCash). This variable contains the list of temporary symbols for cash flow adjustments for management reporting. This list must match rDVTotCFAdj in length. When adjusting a consolidated entity, these symbols will contain the adjustments from the subs within the consolidation.
4. Update the adjustment range variable for adjustments within discontinued operations. (rDVTotDOAdj). Use a pipe symbol to separate the symbol names (e.g. DISCOPS_CFReclass). This variable contains the list of symbols for cash flow adjustments within discontinued operations.
5. Update the adjustment range variable for discontinued operations. (rDVTotDOAdj2). Use a pipe symbol to separate the symbol names (e.g. tDISCOPS_CFReclass). This variable contains the list of temporary symbols for cash flow adjustments for discontinued operations. This list must match rDVTotDOAdj in length. When adjusting a consolidated entity, these symbols will contain the adjustments from the subs within the consolidation.
6. Update the protect range variable for accounts (rAcctsProtect) to include any additional accounts that should not be manually adjusted. Use a pipe symbol to separate the symbol names (e.g. $CFFX$|$CFBeg$|$CFEnd$|$CFCash$)

   > **Note:** The cash flow account symbols in the protect range are defined by the values in the corresponding attributes. For example, $CFFX$ represents the account set in attribute ASCFFX.

7. Click **Save**.

Updating the Post-Selection for additional data view or account symbols

The Post-Selection needs to be updated to include any additional data view symbols to hide if your system is using the summary version of the cash flow calculation.

To update the post-selection for additional data view or account symbols:

1. Double-click on **Pre-Selection**.

2. Update the data views to hide string variable in the summary version block (sDVToHide). Use a pipe symbol to separate the symbol names (e.g. Hide DATAVIEWS, NOCORP_Total|CORP_Total|CONS_Total).
3. You may want to hide some symbols in the detailed version block, if you are using detailed cash flow calculation and there are symbols which do not apply within the data views.

Additional Data Area Definition configuration

The CashFlow data area definition required additional configuration changes to the list of adjustment data views stored in variables rDVTotDOAdj and rDVTotDOAdj2.

To adjust the cash flow data area definition:

1. Double-click on **CashFlow**.
2. Expand the **Additional Configuration section**.
3. For each additional symbol in rDVTotCFAdj2 add a TempSym function attaching the symbol tMGMT, similar to the existing one for tCF_NonCash.

   Note: Any additional TempSym functions need to precede the Sym function attaching MGMT to tMGMT.

4. For each additional symbol in rDVTotDOAdj2 add a TempSym function attaching the symbol tSTATUTORY similar to the existing one for tDISCOPS_CFReclass.

   Note: Any additional TempSym functions need to precede the Sym function attaching DISCOPS_Total to tSTATUTORY.

Implementing Allocations

The Solutions Framework includes a starter kit for implementing allocations, deployed as a data table input app named 'Manage Allocations'. While most of the functionality related to allocations is contained in the Solutions Framework code libraries, the data table and manage allocations app are fully customizable.

Allocations is implemented with the following components:

- Data Table LVAPP_AREAS, deployed during install/upgrade.
- Data Table LVAPP_ALLOCRUN, deployed during install/upgrade.
- Data Table APP_ALLOC, deployed during implementation.
- AREA code library, refer to Longview Developer's Guide for details.
- ALLOC code library, refer to Longview Developer's Guide for details.
- Example Apps: Manage Allocations, Export Allocations, and Import Allocations.
- App: Run Allocations.
- Event rule, deployed during install/upgrade.
- Allocation event, deployed during install/upgrade.

The APP_ALLOC table must be created during implementation as this table can be customized to suit the customer. For more details, see Creating the APP_ALLOC data table.

The Manage Allocations app is provided as an example and can be duplicated and edited to suit the customer. For more details, see The example manage allocations app.

The Export Allocations app is provided as an example and can be duplicated and edited to suit the customer. For more details, see The example export allocations app.

The Import Allocations app is provided as an example and can be duplicated and edited to suit the customer. For more details, see The example import allocations app.

LVAPP_AREAS Table - Overview

The LVAPP_AREAS table is used for storing data area definitions for the allocations in your system.

The LVAPP_AREAS table consists of the following columns:

Column Name	Type	Purpose
ID	NUMBER	Unique identifier for the data table row.
RELID	NUMBER	Unique identifier for the allocation definition. For example, each allocation contains multiple data areas for different purposes. This column will provide the allocation its own specific ID.

Column Name	Type	Purpose
TYPE	STRING	Denotes the purpose of the data area. For example, in the Allocations module the types are: ■ ALLOCSOURCE ■ ALLOCTARGET ■ ALLOCPATTERN ■ ALLOCSOURCEEX ■ ALLOCTARGETEX ■ ALLOCSOURCEOF **Note:** If using the LVAPP_AREAS table for custom apps, it is best practice to prefix your TYPE values for easier filtering. For example, all allocation types are prefixed with “ALLOC”.
DIMENSION	DIMENSION	Stores the dimension for the data area row.
ITEMNO	NUMBER	Stores a unique identifier for each symbol in an area within the same dimension. This would be a value other than 1 only in cases where multiple symbols have been selected for one data area in the same dimension.
SYMBOL	SYMBOL	Stores the symbol for the data area row, based on the dimension in the DIMENSION column.

LVAPP_ALLOCRUN Table - Overview

The LVAPP_ALLOCRUN table is used for storing allocations that have been triggered to run. Once an allocation has completed processing, it is removed from the LVAPP_ALLOCRUN table.

The LVAPP_ALLOCRUN table consists of the following columns:

Column Name	Type	Purpose
ID	NUMBER	Unique identifier for the data table row.
ALLOCID	NUMBER	Unique identifier for the allocation definition.
ALLOCPERIOD	SYMBOL	Stores the time period symbol the allocation was triggered to run on. **Note:** If an allocation was triggered on a parent time period, an individual row for each leaf time period under the selected parent would be created in the LVAPP_ALLOCRUN table.

Column Name	Type	Purpose
PROCESSORDER	NUMBER	Stores the numeric value to define the order in which the allocations will be processed if multiple allocations are selected to run at the same time. Allocations will be processed in ascending order, ensuring each allocation has been submitted and rolled up before the next allocation is run.

Creating the APP_ALLOC data table

The Solutions Framework contains scripts for defining the APP_ALLOC data table. It is filled out with the standard columns but can be modified as required to suit the customer's needs. You will find the SQL script file in the `DataServerRepositories\<Longview Identifier>` within the installation folder.

There are two flavors of the file, one that contains the TMEPERIOD column and one that does not. Use the one containing TIMEPERIOD if you want to vary your allocation definitions by time period. The files will be named:

- SQL Server:
  - APP_ALLOC_SQLServer.sql
  - APP_ALLOC_with_TIME_SQLServer.sql
- Oracle:
  - APP_ALLOC_Oracle.sql
  - APP_ALLOC_with_TIME_Oracle.sql

The following columns are required for the table to be compliant with the Solutions Framework:

- ID
- ACTIVE
- CATEGORY
- DESCRIPTION
- MARKUPPCT
- ALLOCTYPE
- ALLOCPCT
- PROCESSORDER
- USESOURCEOF
- SOURCE
- SOURCEEX

- SOURCEOF
- TARGET
- TARGETEX
- PATTERN

Run Allocations App

Run Allocations is a standard out of box app that allows you to trigger active allocations in your system.

For more information on the Run Allocations app, see the “Run Allocations” section in the *Solutions Framework Administrator Guide*.

Allocation Event Rules

Event rules are deployed during an install/upgrade. These rules are used in conjunction with the Run Allocations App to manually trigger the allocation event.

Event Rules - Longview Transfer Pricing

For a Longview Transfer Pricing deployment, there is one rule created to run the allocations:

5700000

```
KLX(EVNT_5700000###, #ALL, DIM2SET, DIM3SET, DIM4SET, DIM5SET, DIM6SET,
DIM7SET)=000:RUNPROC("applications\Solutions\Events\ALLOC\Event.lvpro")
```

Event Rules - Longview Close

For a Longview Close deployment, there is one rule created for each of the following time periods, with all non-standard dimensions symbols set to <DIMENSION>_ Default in the event rule.

- Actual_PA###
- Forecast_PA###
- Budget_PA###

5400000

```
KLX(EVNT_5400000###, Actual_PA###, ENTITIES_Default, DATAVIEWS_Default,
SCENARIOS_Default, CURRENCIES_Default)=000:RUNPROC
("applications\Solutions\Events\ALLOC\Event.lvpro")
```

5400001

```
KLX(EVNT_5400000###, Forecast_PA###, ENTITIES_Default, DATAVIEWS_Default,
SCENARIOS_Default, CURRENCIES_Default)=000:RUNPROC
("applications\Solutions\Events\ALLOC\Event.lvpro")
```

5400002

```
KLX(EVNT_5400000###, Budget_PA###, ENTITIES_Default, DATAVIEWS_Default,
SCENARIOS_Default, CURRENCIES_Default)=000:RUNPROC
("applications\Solutions\Events\ALLOC\Event.lvpro")
```

Allocation Event

The allocation event (ALLOC) is deployed during an install/upgrade.

The allocation event will perform the following:

1. Retrieve the source data area.
2. Transfer the pattern to the target data area.
3. Apply any target exceptions.
4. Calculate the pattern ratio.
5. Calculate the allocation values.
6. Calculate the source offset value.

Retrieving the Source Data Area

The allocation event will:

1. Download the source data area.
2. Zero out any source exceptions.
3. Transfer the source data area to the source offset data area (for processing in a later stage of the event).
4. The source value is calculated as:

   `(Source Value) * (100% + Markup %) * (Allocation %)`

Note:

- If the source symbol is a parent and the source symbol is the same as the target symbol - the source value will be calculated for each intersection.
- The source, source offset and source exception data areas are defined in the Manage Allocations App and stored in the LVAPP_AREAS table.
- If you are using parallel allocation execution, you may have multiple allocation events executing and each will run for a separate account under the EVNT_symbol. If allocations are triggered for multiple time periods, these will execute in parallel if parallel event rules have been setup. See Configuring Parallel Allocation Execution.

Transferring the pattern to the target area

The pattern is transferred from the defined pattern data area to the defined target data area.

Note:

- This step will be omitted for any allocations of type “Direct Charge” as there is no pattern defined.
- If the target symbol is a parent and the pattern symbol is not the same as the target symbol, the pattern will be copied to each individual symbol below the parent.
- The target and pattern data areas are defined in the Manage Allocations App and stored in the LVAPP_AREAS table.

Applying any target exceptions

Once the pattern has been transferred to the target area, any target exceptions defined will be zeroed out.

Note: This step will be omitted for any allocations of type “Rate” or ”Direct Charge”

A rollup of the target data area will be performed to calculate the new pattern total taking into account any exceptions.

Note: The target exception data area is defined in the Manage Allocations App and stored in the LVAPP_AREAS table.

Calculating the pattern ratio

The pattern ratio is calculated by dividing the pattern that has been transferred by the pattern denominator.

Note: This step will be omitted for any allocations of type “Rate”, as it is assumed the values are already stored as percentages. It will also be omitted for any allocation of type “Direct Charge” as there is no pattern set.

The pattern denominator value will be defined by an intersection, but the symbols for each dimension in the intersection will vary depending on the following conditions:

Condition	Denominator
Target symbol is a parent AND Pattern symbol is not the same as the target symbol	Pattern symbol will be used
Target symbol is a parent AND Pattern symbol is the same as the target symbol	Target symbol will be used
Target symbol is not a parent	Target symbol will be used

Calculating the allocation values

For allocations of type Pattern or Rate, the target value is than calculated by multiplying the source values by the pattern values or rate values. For allocations of type Direct Charge, the target value is than calculated by transferring the source value to the target.

Calculating the source offset value

The source data area is transferred to the source offset data area. The source offset is used to zero out the source value allocated so that allocated values do not change the original consolidated values.

The source offset is calculated by the following formula:

```
(Source Value) * (-1) * (100% + Markup %) * (Allocation %)
```

Note:

- The source offset is optional. If no source offset is defined, this step will be omitted.
- The source and source offset data areas are defined in the Manage Allocations App and stored in the LVAPP_AREAS table.

Configuring Parallel Allocation Execution

Parallel allocation execution allows you to configure allocations to trigger as parallel event triggers rather than executing all allocation in sequence in a single event trigger.

To enable parallel allocation execution

1. Launch Application Administrator.
2. Select **Tools** > **Parallel Event Rule Setup** from the menu.

3. Add a line for the allocation event id and the maximum number of parallel executions.
   - For Tax and Transfer Pricing systems: 5700000, <max> (0 to 32)
   - For Other systems 5400000, <max> (0 to 32) (0 or 1 means single execution only)

   Note: For more details see “Specifying Parallel Event rules” in the *Longview Application Administrator Guide*.

4. In the Server Explorer navigate to **Attributes** > **SYSTEM**.
5. Optionally, locate the attribute `ASAllocParallel` and set its value to TRUE.
6. Optionally, locate the attribute `ASAllocParallelUseCategory` and set its value to TRUE. This will allow you to use the category value of allocations to group related dependant allocations together.
7. Restart the Longview server to enable parallel event execution.

Parallel allocation execution rules

The following rules apply to parallel allocation execution:

Parallel rule setup	Parallel	Use Category	Execution
No, 0 or 1	N/A	N/A	All allocations run in sequence (each period triggered independently)
Yes, 2 - 32	FALSE	FALSE	Allocations run in sequence with allocations in different time periods running in parallel
Yes, 2 - 32	TRUE	FALSE	Each allocation with processing order = 0 runs in parallel (each period triggered independently) All allocations with processing order <> 0 run in sequence (each period triggered independently)
Yes, 2 - 32	TRUE	TRUE	Each allocation with processing order = 0 runs in parallel (each period triggered independently) All allocations with processing order <> 0 in the same category run in sequence (each period triggered independently, and category triggered independently)

Note: Longview Close is configured by default to run allocations in Actual, Budget and Forecast time periods as separate events. Enabling parallel allocation execution will further sub-divide these into additional parallel executions.

The Manage Allocations App

The Solutions Framework contains an example manage allocations app, that is always up to date with the latest changes. Use this app to create an initial Manage Allocations app, or to compare an existing Manage Allocations app to the latest example.

To start configuration of the Manage Allocations app:

1. Select the **Design** module.
2. In the Apps category, expand **Data Collection**.
3. Expand **Data Table Input**.
4. Expand **Existing Apps**.
5. Click on **Manage Allocations** to open it.
6. Click the **Customize** button.
7. Click **Save**.

Modifying the manage allocations app

Manage Allocations allows the user to define allocations to be executed within the system. Allocations are used to spread data from a source to a target based on a pattern or rate.

For example, an allocation may spread IT costs to all departments based on the number of computers in a department. Manage Allocations is implemented as a data table input app and can be modified in the Apps category of the Design tab of the client. Allocation definitions are stored in the APP_ALLOC table. The definitions of the areas are stored in the LVAPP_AREAS table. The EditProcedure function is used to provide a custom cell editor for the definition of areas to avoid having one symbol column per dimension per area. The ALLOC and AREA code libraries are used to handle the editing and storing of area definitions.

> **Note:** In order to use the Manage Allocations app, allocation target symbols must first be created. For more information, see “Managing Allocation Targets” in the *Solutions Framework Administrators Guide*.

Pre-selection

This section checks that allocation symbols have been created prior to running the app. By default, this checks the symbol specified in the system attribute ASAllocRuleParent in the dimension defined in the system attribute ASAllocRuleDim.

- For Close, this is by default the CORP_Alloc symbol in the DATAVIEWS dimension
- For Transfer Pricing, this is by default the LTP_Alloc symbol in the CONTROLS dimension

This section will also check whether the system type that the app is being run on is Tax (Longview Transfer Pricing) or Close and set the following logic and variables as required:

- The list of categories an allocation can be assigned to.

  Note: If you are using parallel allocation execution, you can use categories to group dependant allocations together using categories.

- The list of columns to exclude when a row is duplicated.
- If a TIMEPERIOD column exists in the APP_ALLOC table, the app is set up for time period specific allocations.

Post-selection

If a TIMEPERIOD column exists in the APP_ALLOC table, the left title text is set up to display the selected timeperiod in the Manage Allocations table view.

Data Table Definitions

The app contains two data table definitions: Areas and Alloc. Areas queries the standard LVAPP_ AREAS table and should not be modified. Alloc queries the customer specific APP_ALLOC table and will require modification if the columns are different from the default supplied in the starter kit.

Note: In the Alloc data table definition, the TIMEPERIOD column should be selected. It will be deselected by default.

The additional configuration section contains a time period filter. This will limit the allocations to manage to the time period selected, and only applies to Longview Transfer Pricing implementations that contain a TIMEPERIOD column in the APP_ALLOC table.

Table Views

A single table view is included for the Alloc data table. Modifications will be required if the columns in the Alloc data table are different from the default supplied in the starter kit.

Note: In the Alloc table view, the TIMEPERIOD column should be deselected.

The additional configuration section contains the EditProcedure settings for each of the area columns, and the columns that are excluded when a row is duplicated. It also defines the number of decimals for ID and PROCESSORDER to be 0.

View Actions

A view action is included to run the Allocation Proof app. Running the Allocation proof app allows you to view what the results of the allocation will look like before running the allocation.

The Allocation Proof app was introduced in version 10.3(Build 967). If you have upgraded from an earlier version, the following view action must be created in order to be able to run the allocation proof app:

Name	AllocationProof
Action Location	Row
Views to add actions to	Alloc
Action text	Run Allocation Proof
Action icon	<Leave blank>
Action tooltip text	<Leave blank>
Row restriction expression	<Leave blank>
Action	Run PROCEDURE “ALLOC\RunAllocationProof.lvpro”

Dynamic Actions

Dynamic actions are included to handle when an allocation is added, deleted or duplicated.

Add

The add procedure handles a new allocation being added.

The following steps are performed:

1. The next allocation ID is determined.
2. The ID column is updated with the allocation ID.
3. The allocation ID is passed on to the Edit procedure via the AREA_RELID variable.
4. The markup percentage and allocation percentage are initialized to 0 and 100 respectively.
5. Procedure Edit is executed.

Duplicate

The duplicate procedure handles when an allocation is duplicated.

The following steps are performed:

1. The next allocation ID is determined.
2. The ID column is updated with the allocation ID.
3. The allocation ID is stored in variable AREA_TargetID.
4. The allocation ID of the source allocation is retrieved to the variable AREA_RELID.
5. The areas are duplicated by calling procedure AREA\Duplicate.

Delete

The delete procedure handles an allocation being deleted.

The following steps are performed:

1. Retrieve the allocation ID and set the AREA_RELID.
2. Delete the areas configured for the allocation by calling procedure AREA\DeleteRow.

ProcessingOrder

The ProcessingOrder procedure dynamically validates the value input for Processing Order.

The following steps are performed:

1. The value for processing order is validated to ensure it is not a negative number.
2. If the processing order is a positive number, it is rounded to zero decimal places.

MarkupValidation

The MarkupValidation procedure dynamically validates the value input for Markup %.

The following steps are performed:

1. The value for Markup % is rounded to two decimal places.

AllocPctValidation

The AllocPctValidation procedure dynamically validates the value input for Allocation %:

The following steps are performed:

1. The value for processing order is validated to ensure it is not a negative number and that it is not greater than 100.
2. If the processing order is a positive number and not greater than 100, it is rounded to two decimal places.

Edit Actions

Edit actions provide custom editors for the area columns of the allocations.

Set Area

The set area procedure is not called directly by the user, but called by each edit procedure to update the area displayed in the allocation table. This procedure uses the column index passed in via the colIdx variable to set the value displayed in the allocation table. The area is displayed as comma separated list of symbols with a semicolon to separate dimensions.

Set Source

The set source procedure initializes variables required to edit the source for an allocation. Once the source is updated any required modifications to the source exceptions are applied. Source exceptions are modified when an exception is no longer valid (source changes to a leaf, a source exception is not contained within the selected source parent).

The following variables must be set:

```
AREA_Type = ALLOCSOURCE
```

For each dimension

- ALLOC_<Dimension> = List of symbol specifications to determine valid selections.
- Any dimensions not specified will allow the user to select from all symbols
- If a single symbol is specified, no symbol selector will display on the form for that dimension.

Set Target

The set target procedure initializes variables required to edit the target for an allocation. Once the target is updated any required modifications to the target exceptions are applied. Target exceptions are modified when an exception is no longer valid (target changes to a leaf, a target exception is not contained within the selected target parent).

The following variables must be set:

```
AREA_Type = ALLOCTARGET
```

For each dimension

- ALLOC_<Dimension> = List of symbol specifications to determine valid selections.
- Any dimensions not specified will allow the user to select from all symbols.
- If a single symbol is specified, no symbol selector will display on the form for that dimension.

Set Pattern

The set pattern procedure initializes variables required to edit the pattern for an allocation.

The following variables must be set:

```
AREA_Type = ALLOCPATTERN
```

For each dimension

- ALLOC_<Dimension> = List of symbol specifications to determine valid selections.
- Any dimensions not specified will allow the user to select from all symbols.
- If a single symbol is specified, no symbol selector will display on the form for that dimension.

Set Source Exceptions

The set source exceptions procedure initializes variables required to edit the source exceptions for an allocation. Any dimension in the source for which a parent symbol is selected will allow for exceptions to be specified.

The following variables must be set:

```
AREA_Type = ALLOCSOURCEEX
```

Set Target Exceptions

The set target exceptions procedure initializes variables required to edit the target exceptions for an allocation. Any dimension in the target for which a parent symbol is selected will allow for exceptions to be specified.

The following variables must be set:

```
AREA_Type = ALLOCTARGETEX
```

Set Source Offset

The set source offset procedure initializes variables required to edit the source offset for an allocation.

The following variables must be set:

```
AREA_Type = ALLOCSOURCEOF
```

For each dimension

- ALLOC_<Dimension> = List of symbol specifications to determine valid selections.
- Any dimensions not specified will allow the user to select from all symbols.
- If a single symbol is specified, no symbol selector will display on the form for that dimension.

Note: The valid selections for the offset should normally be a combination of the source and target.

Edit

The Edit procedure allows the user to step through the initial setup of an allocation via a series of forms when a row is added.

Description

The first step of the edit procedure is the description form. The description form is used to set general columns within the allocation definition. Modifications will be required to this procedure and the form for any differences between the default columns.

The columns specified via the Description form are:

- CATEGORY
- DESCRIPTION

- MARKUPPCT
- ALLOCTYPE
- ALLOCPCT
- PROCESSORDER
- USESOURCEOF
- NOTES

Source

The next step is setting the source by calling the SetSource procedure. This procedure is also called when the edit button is clicked in the source field. Once the source is set, the source offset and pattern are also initialized with the symbols selected for the source.

Source Exceptions

The next step is setting the source exceptions by calling the SetSourceExceptions procedure. This procedure is also called when the edit button is clicked in the source exceptions field. Source exceptions are used to include specific symbols from within a hierarchy when the source is a parent. For example, you may want to allocation all selling expenses, except for commissions. Source exceptions are optional and not required for every allocation.

Target

The next step is setting the target by calling the SetTarget procedure. This procedure is also called when the edit button is clicked in the target field.

Once the target is set the following adjustments are made:

1. The pattern is updated to use the entity selected for the target.
2. The source offset is updated to use the allocation symbol and currency selected for the target.

Note: For Longview Close and Plan, the edit procedure assumes the DATAVIEWS dimension contains the target allocation symbols. For Longview Transfer Pricing, the edit procedure assumes the CONTROLS dimension contains the target allocation symbols.

Target Exceptions

The next step is setting the target exceptions by calling the SetTargetExceptions procedure. This procedure is also called when the edit button is clicked in the target exceptions field. Target exceptions cannot be set for an allocation of type “Rate”. Target exceptions are used to exclude specific symbols within a hierarchy if the target is a parent. For example, you may want to exclude the source entity from receiving any of the allocated selling expenses. Target exceptions are optional and not required for every allocation.

Pattern

The next step is setting the pattern by calling the SetPattern procedure. This procedure is also called when the edit button is clicked in the pattern field. The pattern is used to define how the source is allocated to that target. Patterns are not required for Direct Charge allocation types.

Source Offset

The final step is setting the source offset by calling the SetSourceOffset procedure. This procedure is also called when the edit button is clicked in the source offset field. The source offset is used to zero out the source value allocated so that allocated values to not change the original consolidated values. The source offset cannot be set when “Use Source Offset” is unchecked for an allocation.

Post-Refresh

In the post refresh section, the allocation code library is initialized. The library is initialized here because the areas data table must be created prior to the initialization process.

Calculations

The following calculation procedures are run:

Calculation	Run when...	Purpose
UpdateAreas	Data is refreshed	Populate all the area columns in the manage allocations app with data from the LVAPP_AREAS table. This applies to the following columns: ■ Source ■ Source Exceptions ■ Source Offset ■ Target ■ Target Exceptions ■ Pattern

Calculation	Run when...	Purpose
WriteAll	Before data is submitted	Populate the LVAPP_AREAS table with data from the following area columns: ■ Source ■ Source Exceptions ■ Source Offset ■ Target ■ Target Exceptions ■ Pattern

Validations

The validations procedure that will run all the standard allocation validations and provide a datatable output of any validations that did not pass. If there are any custom validations required, this procedure would need to be modified accordingly.

Forms

The description form is used to capture the general columns of the allocation definition. Modification will be required if the columns in the Allocations data table are different from the default supplied in the starter kit.

Publishing the manage allocations app

The manage allocations app should be published to the Calculations category.

To publish the manage allocations app:

1. Select the Design module.
2. In the Administration category, select Apps Publisher.
3. From the list of Apps, select Manage Allocations.
4. From the list of User Groups, select the user groups that will require access to the manage allocations app.
5. From the list of Categories, select MTCECALC - Maintenance - Calculations.
6. The app should now be accessible from the Administration category of the Tax, Close, Plan or Transfer Pricing modules.

The Example Export Allocations App

The Solutions Framework contains an example Export Allocations app, that is always up to date with the latest changes. Use this app to create an initial Export Allocations app, or to compare an existing Export Allocations app to the latest example.

To start configuration of the Export Allocations app:

1. Select the **Design** module.
2. In the Apps category, expand **Configuration**.
3. Expand **Free Form**.
4. Expand **Existing Apps**.
5. Click on **Export Allocations** to open it.
6. Click the **Customize** button.
7. Click **Save**.

Modifying the Export Allocations app

Export Allocations allows the user to export all allocations within the system to a .csv file. If time period specific allocations are being used, allocations can be exported for a selected time period

Export Allocations is implemented as a free form app and can be modified in the Apps category of the Design tab of the client.

Execute Procedure

This is the controlling procedure for exporting allocations. The following steps are performed:

- The export initialization procedure is run.
- The allocation initialization procedure is run
- A time period selection form is displayed if time period specific allocations are being used.
- A filter for the allocations to be downloaded by timeperiod is set up if time period specific allocations are being used.
- The Generate Documents procedure is run to create the export file and datatable definition file based on the columns in the APP_ALLOC table.
- The datatable dtAllocations is created and downloaded containing all allocations stored in the APP_ALLOC database table. If time period specific allocations are being used, only the allocations related to the time period selected will be downloaded.
- The export file path and file name are set. By default, the export file name is Allocations.csv and will be exported to the <LID>\logs\Export Allocations\YYYYMMDD_HHMMSS folder. This can be changed by updating the EXP.FileName property.
- The export confirmation caption and message variables are set. This is the text displayed on the form once the export is complete.
- The export complete form is displayed.

Export Specs - Script

The app contains an AllocationsH (Header), AllocationsF (Footer) and AllocationsSetDefine (for the Set and Define statements) of the allocation export spec. These are used in the Generate Documents procedure to build the export spec file that defines what is exported from the allocations table. All allocation columns are exported except for the attachments as they cannot be exported to a .csv file.

Forms

The SelectTimeperiod form is used to display a time period selection for export, if time period specific allocations are being used.

Export Procedure

The app contains an Export Procedure. This procedure runs the export spec on the dtAllocations table.

Generate Documents Procedure

This procedure creates both the export file and datatable definition file based on the columns in the APP_ALLOC table.

Table Definition - Script

The app contains the Allocations data table definition. This queries the APP_ALLOC table. The table definition will include a time period filter if time period specific allocations are being used.

Publishing the export allocations app

The export allocations app should be published to the Import and Export category.

To publish the export allocations app:

1. Select the **Design** module.
2. In the **Administration** category, select Apps Publisher.
3. From the list of Apps, select **Export Allocations**.
4. From the list of User Groups, select the **user groups** that will require access to the export allocations app.
5. From the list of **Categories**, select **MTCEIMPEXP - Maintenance - Import and Export**.

The app should now be accessible from the Administration category of the Tax, Close, Plan or Transfer Pricing modules.

The Example Import Allocations App

The Solutions Framework contains an example Import Allocations app, that is always up to date with the latest changes. Use this app to create an initial Import Allocations app, or to compare an existing Import Allocations app to the latest example.

To start configuration of the Import Allocations app:

1. Select the **Design** module.
2. In the Apps category, expand **Configuration**.

3. Expand **Free Form**.
4. Expand **Existing Apps**.
5. Click on **Import Allocations** to open it.
6. Click the **Customize** button.
7. Click **Save**.

Modifying the Import Allocations app

Import Allocations allows the user to import allocations into the system from a .csv file. Allocations currently in the system can either be replaced by the contents of the .csv import file or they can be appended to. If time period specific allocations are being used, allocations can be imported to a selected time period. Import Allocations is implemented as a free form app and can be modified in the Apps category of the Design tab of the client.

Execute Procedure

This is the controlling procedure for importing allocations. The following steps are performed:

- The import initialization procedure is run.
- The view initialization procedure is run.
- The validations initialization procedure is run.
- The allocation initialization procedure is run.
- The C_CONTEXT variable is created if it does not already exist.
- The dtAllocations table is added to the VIEW_DataList variable.
- A form is displayed to select the file to import. If time period specific allocations are being used, a time period selection is also included in the form.
- The import procedure is run.

Note: The default contents of the execute procedure should not be modified as they are required for certain code library processing. Additional code can be added to the execute procedure as required.

Import Form

The Import form is used to allow the user to select a file to import. The form also contains a checkbox to select whether allocations are going to be replaced or appended to. If time period specific allocations are being used, the SelectTimePeriod form is also included in the import form.

Import Specs - Script

The app contains an AllocationsH (Header), AllocationsF (Footer) and AllocationsSetDefine (for the Set and Define statements) of the allocation import spec. These are used in the Generate Documents

procedure to build the import spec file that defines what will be imported to the allocations table. All allocation columns are imported except for the following:

- ID: This is auto generated by the app.
- Attachments: Attachments cannot be imported.
- Timeperiod: If using time period specific allocations, the timeperiod in the import file is ignored. It is replaced by the time period selected in the import form.

Import Procedure

The app contains an Import Procedure. The following steps are performed:

- The ALLOC\GetNextAllocID procedure is run to determine the next available allocation ID in the database.
- The Generate Documents procedure is run to create the import file and datatable definition file based on the columns in the APP_ALLOC table.
- The datatable dtAllocations is created and downloaded containing all the allocations stored in the APP_ALLOC database table. If time period specific allocations are being used, only the allocations related to the time period selected will be downloaded.
- If time period specific allocations are being used, a list of RELID's from the dtAllocations table is stored in order to define the list of rows to delete from the dtAreas table if required.
- The datatable dtAreas is created and downloaded containing all the area information stored in the LVAPP_AREAS database table.
- If "Replace allocations" was selected from the Import form, the contents of the dtAllocations table is cleared. Any allocations cleared from the dtAllocations table will also be cleared from the dtAreas table.
- The import is run to import the .csv file contents to the dtAllocations datatable
- The dtAllocations table is than processed to:
  - Determine the correct ID's to assign to the new allocations.
  - For each allocation area column, the ALLOC\Retrieve ToObjectFromString procedure is run to convert the semi colon delimited area lists to area records in the LVAPP_Areas table. The standard allocation area columns include:

- Source
- Source Exceptions
- Source Offset
- Target

- Target Exceptions
- Pattern
- The VIEW\OnSave procedure is run. This will call the following two procedures described below:
  - Validate
  - Save
- If there are any validation errors, a message will be displayed informing the user that no allocations have been imported.

Validate Procedure

The app contains a validate procedure that will run all the standard allocation validations and provide a datatable output of any validations that did not pass. If there are any custom validations required, this procedure would need to be modified accordingly.

Save Procedure

The app contains a save procedure that will upload the allocations if all validations have passed. A message will be displayed informing the user that allocations have been successfully imported.

Generate Documents Procedure

This procedure creates both the import file and datatable definition file based on the columns in the APP_ALLOC table.

Persistent Table Definition

The app contains the Areas data table definition. This queries the LVAPP_AREAS table and will not require any modification.

Table Definition - Script

The app contains the Allocations data table definition. This queries the APP_ALLOC table. The table definition will include a time period filter if time period specific allocations are being used.

Publishing the import allocations app

The import allocations app should be published to the Import and Export category.

To publish the import allocations app:

1. Select the Design module.
2. In the Administration category, select Apps Publisher.
3. From the list of Apps, select Import Allocations.
4. From the list of User Groups, select the user groups that will require access to the import allocations app.
5. From the list of Categories, select MTCEIMPEXP - Maintenance - Import and Export.
6. The app should now be accessible from the Administration category of the Tax, Close, Plan or Transfer Pricing modules.

Customizing Calculations

Solutions Framework includes preconfigured calculation that you can also customize to suit your company's needs.

Customizing pre-configured calculations

Solutions framework provides preconfigured calculations archived in the Solutions\Events folder. Customizations are implemented using Configuration libraries in designer.You can also use this technique to enable debugging options.

Customizing events

If you have a custom modification, you must create a configuration library (via Designer with the custom code (using the same name as the pre-configured code files). The name of the configuration library must be the same name as the event.

Extracting event code (for Solutions events)

To extract event code:

1. On your server, navigate to the \bin folder in your Data Server working directory. For example, `C:\Longview\DataServers\LongviewTax\bin`.
2. Copy the `kar.exe` file to a location on your local computer.
3. Navigate to the `\applications\Solutions\Events\` folder in your Data Server working directory. For example, `C:\Longview\DataServers\LongviewTax\applications\Solutions\Events`.
4. Copy the .kar file for the event you wish to customize. to the same location as the kar.exe file on your local computer.
5. In the local folder, create a Windows Batch file (.bat) with the following command:

   ```
   kar x EventName.kar
   ```

   where `EventName.kar` is the kar file for the event that you want to customize code for.

   > **Note:** To list all the files and folders in the .kar file, type kar t EventName.kar.

6. Double-click the batch file. The command extracts the folders from the EventName.kar file into the local folder. In addition, there will be an EventName folder created that can be ignored.

   > **Note:** To make it easier to find documents you can delete all files with the extension ".xml" as these files are used internally by Designer only.

   Once you have extracted the pre-configured code, you can modify it and then make it available for your system.

Customizing event code (for Solutions events)

To enable debugging options:

1. In Designer create a new configuration library named **EventName**.
2. Create a procedure named **SetDebugOptions**.
3. Add the following lines:
   a. Run PROCEDURE "DEBUG\Init.lvpro"
   b. Set VARIABLE DEBUG_Mode = 1
4. Optionally add the following:
   a. Set VARIABLE DEBUG_Messaging = 1, to add additional debug messages to the log.
   b. Set VARIABLE DEBUG_Profile = 1, to add execution profile statistics to the log.
   c. Set VARIABLE DEBUG_HistoryDetail = 1, to enable execution history.
   d. Set VARIABLE DEBUG_Timerrun = 1, to enable the execution history timer.
5. Save the configuration library.

To customize event code:

1. In Designer create a new configuration library named EventName.
2. Create the document you want to customize in the configuration library with the same name as the extracted document.
3. Right-click the source file you want to customize and open it in your preferred text editor.
4. Copy the contents of the entire document and paste the contents to the document created in Designer.
5. Make the necessary modifications, then save the configuration library.

Creating a custom calculation method in Longview Tax

Instead of using the available standard calculations for calculating tax amounts, it is possible to define the following custom calculation methods to run net income before tax or book-tax difference accounts:

- Creating a custom calculation method for net income before tax accounts
- Creating a custom calculation method for book-tax difference accounts

Creating a custom calculation method for net income before tax accounts

You can use Tax Provision to define a custom calculation method to automatically calculate values in the current provision (current tax charge) for NIBT accounts.

To create a custom calculation method for NIBT accounts:

1. Specify a name for the custom calculation method in NIBT Automation - Custom Actuals Calculation Methods in the System_Settings.csv import file. For more information, see Preparing an Import File for System Settings.

   > **Note:** The name for the new custom calculation method will display in the Net Income Before Tax editor in the Actuals Calculation Method list and will also be the required name for the calculation procedure document. Choose a meaningful name for the calculation method, but make sure that it is of a reasonable length for the drop-down list and that it conforms to the character restrictions for procedure names.

2. Do one of the following depending on your system type:

   - For an ASC system, add a new procedure in the APPFRMWRK\Events\NIBTTransfer\Code directory with the name NIBTTransfer_name.lvpro.
   - For an IAS system, add a new procedure in the APPFRMWRK\Events\NIBTTransfer12\Code directory with the name NIBTTransfer_name.lvpro.

     Where name is the name of the custom calculation method as set for NIBT Automation - Custom Actuals Calculation Methods in the System_Settings.csv import file.

3. Add the required calculation logic to the new procedure to perform the custom calculation of the current provision (current tax charge) amounts.
4. Save the procedure in the directory.
5. Sign in to Tax Provision.
6. In the Tax Provision navigation pane, click the Administration category.
7. Expand Accounts, expand Net Income Before Tax, and then click Net Income Before Tax. The Net Income Before Tax editor opens.
8. Select Automate NIBT Amounts for the required NIBT accounts.
9. For Actuals Calculation Method, select the new custom calculation method.
10. Make the appropriate selections in the remaining fields to define the percentage and the source symbols required for calculation. For more information, see the *Longview Tax Administrator's Guide.*
11. Click Save. Whenever the NIBT Transfer calculation is initiated, your custom calculation method runs for the selected NIBT accounts.

Creating a custom calculation method for book-tax difference accounts

You can use Tax Provision to define a custom calculation method to automatically calculate values in the current provision for permanent differences and temporary differences, and in the gross temporary difference rollforward for temporary differences.

To create a custom calculation method for difference accounts:

1. Specify a name for the custom calculation method in Current Tax Automation - Custom Actuals Calculation Methods in the System_Settings.csv import file. For more information, see Preparing an

Import File for System Settings.

> **Note:** The name for the new custom calculation method will display in the Book-Tax Differences editor in the Actuals Calculation Method list and will also be the required name for the calculation procedure document. Choose a meaningful name for the custom calculation method, but make sure that it is of a reasonable length for the drop-down list and that it conforms to the character restrictions for procedure names.

In the following code examples, the name specified for the custom calculation method is Labeled Forecast.

2. Do one of the following depending on your system type:
   - For an ASC system, add a new procedure in the APPFRMWRK\Events\AutoPrmTmp\Code directory with the name AutoPrmTmp_name.lvpro.
   - For an IAS system, add a new procedure in the APPFRMWRK\Events\AutoPrmTmp12\Code directory with the name AutoPrmTmp12_name.lvpro.

     Where name is the name of the custom calculation method as set for Current Tax Automation - Custom Actuals Calculation Methods in the System_Settings.csv import file.
3. Add the required calculation logic to the new procedure to perform the custom calculation of the current tax amounts.
4. Save the procedure in the directory.
5. Update the model to include the appropriate code.
6. Sign in to Longview Tax.
7. In the Tax Provision navigation pane, click the Administration category.
8. Expand Accounts, expand Book-Tax Differences, and then click Book-Tax Differences. The Book-Tax Differences editor opens.
9. Select Automate Current Tax Amounts for the required difference accounts.
10. For Actuals Calculation Method, select the new custom calculation method.
11. Make the appropriate selections in the remaining fields to define the percentage and the source symbols required for calculation. For more information, see the *Longview Tax Administrator's Guide*.
12. Click Save. Whenever the Perm/Temp Automation calculation is initiated, your custom calculation method runs for the selected difference accounts.

Adding custom code to rollover

The Tax Provision rollover code includes custom code hooks. These custom code hooks allow you to add custom code into the rollover to suit your company's specific requirements.

There are three places in the rollover code from which these custom code hooks are run:

- At the end of the locking procedure
- At the end of the copy procedure
- At the end of the rollover code

The following are the .lvpro files that are called during the rollover process. In order for these files to run during rollover, you must create the .lvpro files with the exact names as follows:

File	Rollover Type	Used...
TxRoMAppLck.lvpro	Period End	At the end of the locking procedure; allows for locking of additional hierarchies for all dimensions.
TxRoMAppC.lvpro	Period End	At the end of the copy procedure; allows for additional copies to be added for custom data or the ability to override standard copies.
TxRoMApp.lvpro	Period End	At the end of the rollover code; allows for setting of additional attributes.
TxRoYAppLck.lvpro	Year End	At the end of the locking procedure; allows for locking of additional hierarchies for all dimensions.
TxRoYAppC.lvpro	Year End	At the end of the copy procedure; allows for additional copies to be added for custom data or the ability to override standard copies.
TxRoYApp.lvpro	Year End	At the end of the rollover code; allows for setting of additional attributes.

Caution: Longview Solutions strongly recommends that you test any custom rollover code thoroughly before using it in a live system. Modifying rollover code could result in data loss.

To add custom code using a rollover hook, follow these steps:

1. Locate and open your company's Tax Rollover code folder. For example, `...\DataServers\<LID>\applications\Longview Tax\Tax Rollover\Code`.
2. In the folder located in step 1, create the following .lvpro files:
   - TxRoMAppLck.lvpro
   - TxRoMAppC.lvpro
   - TxRoMApp.lvpro
   - TxRoYAppLck.lvpro
   - TxRoYAppC.lvpro
   - TxRoYApp.lvpro

3. Add custom code as desired.

Note: For more information on customizing code, see Customizing Event Code.

Rule ID conventions for Tax Provision

If you create any custom rules, the rule IDs must between 1 and 1999999. The following rule IDs are reserved for Tax Provision system rules:

Rule type	Area	ID Range
Validation	■ Reserved for Future Use ■ Reserved for Custom Use	■ 1000000-1899999 ■ 1900000-1999999
Model	■ General ■ Common areas ■ ASC ■ IAS ■ ASC - Interim ■ IAS - Interim ■ ASC - MRP ■ IAS - MRP ■ Reserved for Future Use ■ Reserved for Custom Use	■ 2000000–2999999 ■ 2000000–2099999 ■ 2100000–2199999 ■ 2200000–2299999 ■ 2300000–2399999 ■ 2400000–2499999 ■ 2500000-2599999 ■ 2600000-2699999 ■ 2700000-2899999 ■ 2900000-2999999

Rule type	Area	ID Range
Rollup	■ General ■ ASC ■ IAS ■ Reserved for Future Use ■ Reserved for Custom Use	■ 3000000–3999999 ■ 3100000–3199999 ■ 3200000–3299999 ■ 3300000-3899999 ■ 3900000-3999999
Query	■ General ■ Common ■ ASC ■ IAS ■ ASC - MRP ■ IAS - MRP ■ Reserved for Future Use ■ Reserved for Custom Use	■ 4000000–4999999 ■ 4000000–4099999 ■ 4100000–4199999 ■ 4400000–4499999 ■ 4500000-4599999 ■ 4600000-4699999 ■ 4700000-4799999 ■ 4900000-4999999

Rule type	Area	ID Range
Event	■ General ■ ASC ■ IAS ■ LTP ■ P2 ■ Reserved for Future Use ■ Reserved for Custom Use	■ 5000000–5999999 ■ 5100000–5199999 ■ 5500000–5599999 ■ 5700000-5709999 ■ 5750000-5759999 ■ 5800000-5899999 ■ 5900000-5999999